D 8

```
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
```

L 11

```
**FILE**ID**LIBSCOPY
```

```
LL            IIIIII    BBBBBBBB      SSSSSSSS    CCCCCCCC    OOOOOO    PPPPPPPP    YY      YY
LL            IIIIII    BBBBBBBB      SSSSSSSS    CCCCCCCC    OOOOOO    PPPPPPPP    YY      YY
LL              II      BB      BB  SS          CC          OO      OO  PP      PP  YY      YY
LL              II      BB      BB  SS          CC          OO      OO  PP      PP  YY      YY
LL              II      BB      BB  SS          CC          OO      OO  PP      PP    YY  YY
LL              II      BB      BB  SS          CC          OO      OO  PP      PP    YY  YY
LL              II      BBBBBBBB      SSSSSS    CC          OO      OO  PPPPPPPP        YY
LL              II      BBBBBBBB      SSSSSS    CC          OO      OO  PPPPPPPP        YY
LL              II      BB      BB          SS  CC          OO      OO  PP              YY
LL              II      BB      BB          SS  CC          OO      OO  PP              YY
LL              II      BB      BB          SS  CC          OO      OO  PP              YY          ....
LL              II      BB      BB          SS  CC          OO      OO  PP              YY          ....
LLLLLLLLLL    IIIIII    BBBBBBBB    SSSSSSSS      CCCCCCCC    OOOOOO    PP              YY          ....
LLLLLLLLLL    IIIIII    BBBBBBBB    SSSSSSSS      CCCCCCCC    OOOOOO    PP              YY          ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
0 MODULE LIB$SCOPY (
0
0               IDENT = '1-018' ! File: LIBSCOPY.B32    Edit: DG1018
0
0               ) =
1 BEGIN
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1 !
1
1 !++
1 ! FACILITY: General library: string handling
1 !
1 ! ABSTRACT:
1 !
1 !       This module contains routines to allocate and deallocate
1 !       strings.  Also included are the basic string copying routines.
1 !
1 ! ENVIRONMENT:  VAX-11 User Mode
1 !
1 ! ORIGINAL AUTHOR: Tomas N. Hastings, 08-OCT-1977
1 ! REWRITTEN BY: John Sauter, 14-MAR-1979
1 !
1 ! MODIFIED BY:
1 !
1 ! 0-09  - Pass by-reference.  TNH 19-DEC-77
1 ! 0-10  - Change order of arguments to conform to manual.  JMT 5-Mar-78
1 ! 0-11  - Fix typo in PIC case table.  DGP 29-Jun-78
1 ! 0-12  - Change JSB entry point names to DX6.  TNH 5-July-78
1 ! 0-14  - Add dynamic descriptors.  TNH 5-July-78
1 ! 0-28  - Remove entry points not in manual.  TNH 1-Aug-78
1 ! 0-29  - Make REMQUE be PIC.  TNH 2-Aug-78
1 ! 0-31  - And INSQUE be PIC.  TNH 2-Aug-78
1 ! 0-32  - Compute effective adr before REMQUE to be PIC.  TNH 2-Aug-78
1 ! 1-001 - Update version number and copyright notice.  JBS 16-NOV-78
1 ! 1-002 - Add "_" to the PSECT directives.  JBS 21-DEC-78
```

LIB$
1-0

; R

```
;    58         0058   1 ! 1-003 - Make external references use 32-bit addresses. JBS 11-FEB-1979
;    59         0059   1 ! 1-004 - Fix allocation of 1- and 2-character strings so that they
;    60         0060   1 !          don't expand the region unconditionally, but instead use
;    61         0061   1 !          an 8-character string.  JBS 11-FEB-1979
;    62         0062   1 ! 1-005 - Edit 004 introduced a bug: strings just too large for the
;    63         0063   1 !          queues try to allocate from them.  Fix it.  JBS 18-FEB-1979
;    64         0064   1 ! 1-006 - Do a total rewrite: these routines are now maintained
;    65         0065   1 !          just for compatability: they call the STR facility.
;    66         0066   1 !          JBS 20-MAR-1979
;    67         0067   1 ! 1-007 - Call the string facility using input scalars by reference.
;    68         0068   1 !          JBS 18-MAY-1979
;    69         0069   1 ! 1-008 - Change calls to STR$COPY.  JBS 16-JUL-1979
;    70         0070   1 ! 1-009 - LIB$SCOPY_DXDX was incorrectly checking for truncation.
;    71         0071   1 !         DGP 09-Oct-79
;    72         0072   1 ! 1-010 - Make compatible with release 1 return codes.  RW  21-Jan-1980
;    73         0073   1 ! 1-011 - Reorganized string routines to all find their way to
;    74         0074   1 !         LIB$SCOPY_R_DX6.  Introduce new classes of descriptors
;    75         0075   1 !         to be recognized by string copying routines.
;    76         0076   1 !         Change LIB$SFREEN_DD6 to accept count by immediate value,
;    77         0077   1 !         they way its external documentation says it should operate.
;    78         0078   1 !         RKR 27-MAR-1981
;    79         0079   1 ! 1-012 - Correct some comments to more accurately reflect actual code.
;    80         0080   1 !         RKR 24-AUG-1981
;    81         0081   1 ! 1-013 - Correct which error codes are returned in LIB$SCOPY_R_DX6.
;    82         0082   1 !         RKR 4-SEP-1981.
;    83         0083   1 ! 1-014 - Return an error if the caller attempts to deallocate a non-dynamic
;    84         0084   1 !         string.  SBL 9-Sep-1981
;    85         0085   1 ! 1-015 - Add special-case code to process string descriptors that
;    86         0086   1 !         "read" like fixed string descriptors.  RKR 7-OCT-1981.
;    87         0087   1 ! 1-016 - Change code in LIB$SGET1_DD and LIB$SGET1_DD_R6 so that
;    88         0088   1 !         if either is called with a descriptor that is not CLASS_D,
;    89         0089   1 !         no problems will result -- i.e., replace validity check for
;    90         0090   1 !         CLASS_D with code to force CLASS_D.  This is what occurred
;    91         0091   1 !         in VMS RTL V2.x of these routines.  Functionality was
;    92         0092   1 !         inadvertently changed in producing Version 3 routines.
;    93         0093   1 !         In LIB$SCOPY_DXDX6, don't use $STR$GET_LEN_ADDR macro, since
;    94         0094   1 !         this macro uses STR$ANALYZE_SDESC_R1 -- don't want the
;    95         0095   1 !         signaling routine used by a LIB$ routine.
;    96         0096   1 !         Redirect jsb's from LIB$ANALYZE_SDESC_R3 to
;    97         0097   1 !         LIB$ANALZYE_SDESC_R2.
;    98         0098   1 !         Remove checks on contents of desciptors other than insuring
;    99         0099   1 !         that DSC$A_ARSIZE is < 65K for class_A and class_NCA.
;   100         0100   1 !         RKR 18-NOV-1981.
;   101         0101   1 ! 1-017 - Add support for class SO string descriptors.  DG 3-OCT-1983.
;   102         0102   1 ! 1-018 - Change class SO string descriptors to SB.  DG 27-Feb-1984.
;   103         0103   1 !--
;   104         0104   1
;   105         0105   1 !<BLF/PAGE>
;   107          0106  1 !
;   108          0107  1 ! SWITCHES:
;   109          0108  1 !
;   110          0109  1
;   111          0110  1 SWITCHES ADDRESSING_MODE
;   112          0111  1              (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;   113          0112  1
;   114          0113  1 !
;   115          0114  1 ! LINKAGES:
;   116          0115  1 !
;   117          0116  1 REQUIRE 'RTLIN:STRLNK';            ! Linkages
;   118          0301  1
;   119          0302  1 !
;   120          0303  1 ! TABLE OF CONTENTS:
;   121          0304  1 !
;   122          0305  1
;   123          0306  1 FORWARD ROUTINE
;   124          0307  1     LIB$SGET1_DD,                             ! Allocate a string
;   125          0308  1     LIB$SGET1_DD_R6 : STRING_JSB,             ! (JSB entry point)
;   126          0309  1     LIB$SFREE1_DD,                            ! Deallocate a string
;   127          0310  1     LIB$SFREE1_DD6 : STRING_JSB,              ! (JSB entry point)
;   128          0311  1     LIB$SFREEN_DD,                            ! Deallocate N strings
;   129          0312  1     LIB$SFREEN_DD6 : STRING_JSB,              ! (JSB entry point)
;   130          0313  1     LIB$SCOPY_DXDX,                           ! Copy a string by
;   131          0314  1                                               ! descriptor
;   132          0315  1     LIB$SCOPY_DXDX6 : STRING_JSB,             ! (JSB entry point)
;   133          0316  1     LIB$SCOPY_R_DX,                           ! Copy a string by
;   134          0317  1                                               ! reference
;   135          0318  1     LIB$SCOPY_R_DX6 : STRING_JSB;             ! (JSB entry point)
;   136          0319  1
;   137          0320  1 !
;   138          0321  1 ! INCLUDE FILES:
;   139          0322  1 !
;   140          0323  1
;   141          0324  1 REQUIRE 'RTLIN:STRMACROS';         ! String macros
;   142          1240  1 REQUIRE 'RTLIN:RTLPSECT';          ! Macros for defining psects
;   143          1335  1
;   144          1336  1 LIBRARY 'RTLSTARLE';               ! System symbols
;   145          1337  1
;   146          1338  1 !
;   147          1339  1 ! MACROS: NONE
;   148          1340  1 !
;   149          1341  1 !
;   150          1342  1 ! EQUATED SYMBOLS:
;   151          1343  1 !
;   152          1344  1 LITERAL
;   153          1345  1     MAX_SIZE = 65535;              ! Maximum size string
;   154          1346  1
;   155          1347  1 !
;   156          1348  1 ! PSECTS:
;   157          1349  1 !
;   158          1350  1 DECLARE_PSECTS (LIB);              ! Declare psects for LIB$ facility
;   159          1351  1 !
;   160          1352  1 ! OWN STORAGE:
;   161          1353  1 !
;   162          1354  1 !     NONE
;   163          1355  1 !
;   164         1356  1 ! EXTERNAL REFERENCES:
;   165         1357  1 !
;   166         1358  1
;   167         1359  1 EXTERNAL LITERAL
;   168         1360  1     STR$_NORMAL,                ! (Used in macro $STR$DEALLOCATE)
;   169         1361  1     LIB$_FATERRLIB,             ! Fatal error in the library
;   170         1362  1     LIB$_INSVIRMEM,             ! Insufficient virtual memory
;   171         1363  1     LIB$_INVSTRDES,             ! Invalid string descriptor
;   172         1364  1     LIB$_STRTRU;                ! String truncated
;   173         1365  1
;   174         1366  1 EXTERNAL ROUTINE
;   175         1367  1     LIB$ANALYZE_SDESC_R2 : LIB$ANALYZE_SDESC_JSB_LINK ;
```

```
1 GLOBAL ROUTINE LIB$SGET1_DD (                   ! Allocate a dynamic string
1
1               LEN,                              ! Number of bytes to allocate
1               DESCRIP                           ! Descriptor to allocate into
1                                   ) =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !             Allocate a string.  LEN bytes are allocated to DESCRIP, which
1 !             is assumed to be a dynamic descriptor.  If the descriptor
1 !             already has storage allocated to it, but not enough, the old
1 !             storage is deallocated.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !             LEN.rwu.r       The number of bytes to allocate.
1 !             DESCRIP.wqu.r   The descriptor.  The DSC$B_DTYPE field is not
1 !                             touched.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !             NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !             NONE
1 !
1 ! COMPLETION CODES:
1 !
1 !             SS$_NORMAL      All is OK.
1 !             LIB$_INSVIRMEM  There was not enough virtual memory to allocate
1 !                             the string.
1 !             LIB$_FATERRLIB  Fatal error in the library
1 !
1 ! SIDE EFFECTS:
1 !
1 !             May deallocate the descriptor's storage and allocate new
1 !             storage for it.
1 !
1 !--
1
2     BEGIN
2
2     MAP
2         DESCRIP : REF BLOCK [ , BYTE];
2
2 !+
2 ! Deallocate old space (if necesaary) and allocate new space.
2 !-
2     RETURN LIB$SGET1_DD_R6 ((..LEN AND %X'FFFF'), .DESCRIP) ;
1     END;                                          ! end of LIB$SGET1_DD

                                                .TITLE  LIB$SCOPY
                                                .IDENT  \1-018\
                                                        .EXTRN  STR$_NORMAL, LIB$_FATERRLIB
                                                        .EXTRN  LIB$_INSVIRMEM, LIB$_INVSTRDES
                                                        .EXTRN  LIB$_STRTRU, LIB$ANALYZE_SDESC_R2

                                                        .PSECT  _LIB$CODE,NOWRT,  SHR,  PIC,2

                                   007C 00000           .ENTRY  LIB$$GET1_DD, Save R2,R3,R4,R5,R6               ; 1368
               51        08    AC   D0 00002            MOVL    DESCRIP, R1                                     ; 1418
               50        04    BC   3C 00006            MOVZWL  @LEN, R0                                        ;
                             0000V  30 0000A            BSBW    LIB$$GET1_DD_R6                                 ;
                                    04 0000D            RET                                                     ; 1419

; Routine Size:  14 bytes,    Routine Base:  _LIB$CODE + 0000
```

```
1 GLOBAL ROUTINE LIB$SGET1_DD_R6 (         ! Allocate a dynamic string
1
1         LEN,                             ! Number of bytes to allocate
1         DESCRIP                          ! Descriptor to allocate into
1                              ) : STRING_JSB =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Allocate a string.  LEN bytes are allocated to DESCRIP, which
1 !       is assumed to be a dynamic descriptor.  If the descriptor
1 !       already has storage allocated to it, but not enough, the old
1 !       storage is deallocated.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       LEN.rwu.v       The number of bytes to allocate.
1 !       DESCRIP.wqu.r   The descriptor.  The DSC$B_DTYPE field is not
1 !                       touched.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION CODES:
1 !
1 !       SS$_NORMAL      All is OK.
1 !       LIB$_INSVIRMEM  There was not enough virtual memory to allocate
1 !                       the string.
1 !       LIB$_FATERRLIB  Fatal error in the library
1 !
1 ! SIDE EFFECTS:
1 !
1 !       May deallocate the descriptor's storage and allocate new
1 !       storage for it.
1 !
1 !--
1
2     BEGIN
2     LOCAL
2         RETURN_STATUS ;
2
2     MAP
2         DESCRIP : REF BLOCK [, BYTE] ;
2
2 !+
2 ! Make the descriptor be a dynamic string.
2 !-
2     DESCRIP [DSC$B_CLASS] = DSC$K_CLASS_D;
2
2     RETURN_STATUS = SS$_NORMAL ;         ! assume success
2
2 !+
;   287            1477  2 ! see if current space needs to be deallocated and reallocated
;   288            1478  2 !-
;   289          P 1479  3     IF ( $STR$NEED_ALLOC (( .LEN AND %X'FFFF'),
;   290            1480  3                         $STR$DYN_AL_LEN (DESCRIP)))
;   291            1481  2     THEN
;   292            1482  3         BEGIN
;   293            1483  3         !
;   294            1484  3         ! give back old space
;   295            1485  3         !
;   296            1486  4         IF ( RETURN_STATUS = $STR$DEALLOCATE (DESCRIP))
;   297            1487  3         THEN
;   298            1488  3             !
;   299            1489  3             ! and get new space
;   300            1490  3             !
;   301          P 1491  3             RETURN_STATUS = $STR$ALLOCATE (( .LEN AND %X'FFFF'),
;   302            1492  3                                         DESCRIP ) ;
;   303            1493  3         END
;   304            1494  3
;   305            1495  2     ELSE
;   306            1496  2
;   307            1497  2         !
;   308            1498  2         ! old space can be reused
;   309            1499  2         !
;   310            1500  2         $STR$LENGTH (DESCRIP) = (.LEN AND %X'FFFF') ;
;   311            1501  2 !+
;   312            1502  2 ! at this point, RETURN_STATUS contains one of:
;   313            1503  2 !       A.  originally assigned status i.e., SS$_NORMAL
;   314            1504  2 !       B.  failure status from $STR$DEALLOCATE
;   315            1505  2 !       C.  status from $STR$ALLOCATE
;   316            1506  2 !-
;   317            1507  2
;   318            1508  2     RETURN .RETURN_STATUS ;
;   319            1509  1     END ;                           ! of routine LIB$GET1_DD_R6


                                                  .EXTRN  STR$$Q_SHORT_Q, LIB$FREE_VM
                                                  .EXTRN  STR$_FATINTERR, STR$$INIT
                                                  .EXTRN  STR$$V_INIT, STR$$ALOC_SHORT
                                                  .EXTRN  LIB$GET_VM, STR$_INSVIRMEM

               5E            04  C2 00000 LIB$GET1_DD_R6::
                                                  SUBL2   #4, SP                                  ; 1420
               52            51  D0 00003         MOVL    R1, R2                                  :
               54            50  D0 00006         MOVL    R0, R4                                  :
        03     A2            02  90 00009         MOVB    #2, 3(DESCRIP)                          ; 1472
                             01  DD 0000D         PUSHL   #1                                      ; 1474
               53      04    A2  D0 0000F         MOVL    4(DESCRIP), R3                          ; 1480
                             50  D4 00013         CLRL    R0                                      :
                             53  D5 00015         TSTL    R3                                      :
                             06  12 00017         BNEQ    1$                                      :
                             50  D6 00019         INCL    R0                                      :
                             51  D4 0001B         CLRL    R1                                      :
                             13  11 0001D         BRB     3$                                      :
      00F0     8F            62  B1 0001F 1$:     CMPW    (DESCRIP), #240                         :
                             05  1B 00024         BLEQU   2$                                      :
               51            62  3C 00026         MOVZWL  (DESCRIP), R1                           :
```

```
                                        07   11 00029           BRB      3$                                      ;
                         51             53   D0 0002B 2$:       MOVL     R3, STRING_BLOCK                         ;
                         51        FE   A1   3C 0002E           MOVZWL   -2(STRING_BLOCK), R1                     ;
              000000F0   8F             51   D1 00032 3$:       CMPL     R1, #240                                 ;
                                        23   1F 00039           BLSSU    7$                                       ;
                         04             50   E9 0003B           BLBC     R0, 4$                                   ;
                                        51   D4 0003E           CLRL     R1                                       ;
                                        13   11 00040           BRB      6$                                       ;
                  00F0   8F             62   B1 00042 4$:       CMPW     (DESCRIP), #240                          ;
                                        05   1B 00047           BLEQU    5$                                       ;
                         51             62   3C 00049           MOVZWL   (DESCRIP), R1                            ;
                                        07   11 0004C           BRB      6$                                       ;
                         51             53   D0 0004E 5$:       MOVL     R3, STRING_BLOCK                         ;
                         51        FE   A1   3C 00051           MOVZWL   -2(STRING_BLOCK), R1                     ;
   51           54       10             00   ED 00055 6$:       CMPZV    #0, #16, LEN, R1                         ;
                                        23   13 0005A           BEQL     11$                                      ;
                                        24   11 0005C           BRB      12$                                      ;
                         04             50   E9 0005E 7$:       BLBC     R0, 8$                                   ;
                                        51   D4 00061           CLRL     R1                                       ;
                                        13   11 00063           BRB      10$                                      ;
                  00F0   8F             62   B1 00065 8$:       CMPW     (DESCRIP), #240                          ;
                                        05   1B 0006A           BLEQU    9$                                       ;
                         51             62   3C 0006C           MOVZWL   (DESCRIP), R1                            ;
                                        07   11 0006F           BRB      10$                                      ;
                         51             53   D0 00071 9$:       MOVL     R3, STRING_BLOCK                         ;
                         51        FE   A1   3C 00074           MOVZWL   -2(STRING_BLOCK), R1                     ;
   51           54       10             00   ED 00078 10$:      CMPZV    #0, #16, LEN, R1                         ;
                                        03   1A 0007D           BGTRU    12$                                      ;
                                      00CE   31 0007F 11$:      BRW      25$                                      ;
                         50 00000000G   8F   D0 00082 12$:      MOVL     #STR$_NORMAL, RETURN_STATUS              ; 1486
                                        53   D5 00089           TSTL     R3                                       ;
                                        3C   13 0008B           BEQL     14$                                      ;
                  00F0   8F             62   B1 0008D           CMPW     (DESCRIP), #240                          ;
                                        1A   1A 00092           BGTRU    13$                                      ;
                         51             53   D0 00094           MOVL     R3, STRING_BLOCK                         ;
                         51        FE   A1   3C 00097           MOVZWL   -2(STRING_BLOCK), ALLOC_LENGTH           ;
                                        51   D7 0009B           DECL     R1                                       ;
                         51             07   8A 0009D           BICB2    #7, R1                                   ;
                         51 00000000G0041    9E 000A0           MOVAB    STR$$Q_SHORT_Q[R1], INSQUE_ADDR          ;
                    00   B1             63   0E 000A8           INSQUE   (R3), @0(INSQUE_ADDR)                    ;
                                        18   11 000AC           BRB      14$                                      ;
                                   04   A2   9F 000AE 13$:      PUSHAB   4(DESCRIP)                               ;
                    08   AE             62   3C 000B1           MOVZWL   (DESCRIP), 8(SP)                         ;
                                   08   AE   9F 000B5           PUSHAB   8(SP)                                    ;
             00000000G   00             02   FB 000B8           CALLS    #2, LIB$FREE_VM                          ;
                         07             50   E8 000BF           BLBS     RETURN_STATUS, 14$                       ;
                         50 00000000G   8F   D0 000C2           MOVL     #STR$_FATINTERR, RETURN_STATUS           ;
                         6E             50   D0 000C9 14$:      MOVL     RETURN_STATUS, RETURN_STATUS             ;
                         7F             50   E9 000CC           BLBC     RETURN_STATUS, 24$                       ;
                         07 00000000G   00   E8 000CF           BLBS     STR$$V_INIT, 15$                         ; 1492
             00000000G   00             00   FB 000D6           CALLS    #0, STR$$INIT                            ;
                         50 00000000G   8F   D0 000DD 15$:      MOVL     #STR$_NORMAL, RETURN_STATUS              ;
                  00F0   8F             54   B1 000E4           CMPW     LEN, #240                                ;
                                        40   1A 000E9           BGTRU    21$                                      ;
                                        54   B5 000EB           TSTW     LEN                                      ;
                                        04   12 000ED           BNEQ     16$                                      ;
                                        55   D4 000EF           CLRL     TEMP                                     ;
```

```
                               2F  11 000F1         BRB     20$                                         ;
                  51           54  3C 000F3 16$:    MOVZWL  LEN, R1                                     ;
                               51  D7 000F6         DECL    R1                                          ;
                  51           07  8A 000F8         BICB2   #7, R1                                      ;
                  56 00000000G0041  9E 000FB        MOVAB   STR$$Q_SHORT_Q[R1], REMQUE_ADDR             ;
                  55       00  B6  0F 00103 17$:    REMQUE  @0(REMQUE_ADDR), TEMP                       ;
                               05  1D 00107         BVS     18$                                         ;
                  53           01  D0 00109         MOVL    #1, ALLOC_DONE                              ;
                               0C  11 0010C         BRB     19$                                         ;
                               53  D4 0010E 18$:    CLRL    ALLOC_DONE                                  ;
                  7E           54  3C 00110         MOVZWL  LEN, -(SP)                                  ;
        00000000G 00           01  FB 00113         CALLS   #1, STR$$ALOC_SHORT                         ;
                  05           53  E8 0011A 19$:    BLBS    ALLOC_DONE, 20$                             ;
                  2B           50  E9 0011D         BLBC    RETURN_STATUS, 23$                          ;
                               E1  11 00120         BRB     17$                                         ;
                  26           50  E9 00122 20$:    BLBC    RETURN_STATUS, 23$                          ;
              04  A2           55  D0 00125         MOVL    TEMP, 4(DESCRIP)                            ;
                               1D  11 00129         BRB     22$                                         ;
                           04  A2  9F 0012B 21$:    PUSHAB  4(DESCRIP)                                  ;
              08  AE           54  3C 0012E         MOVZWL  LEN, 8(SP)                                  ;
                           08  AE  9F 00132         PUSHAB  8(SP)                                       ;
        00000000G 00           02  FB 00135         CALLS   #2, LIB$GET_VM                              ;
                  09           50  E8 0013C         BLBS    RETURN_STATUS, 22$                          ;
                  50 00000000G 8F  D0 0013F         MOVL    #STR$_INSVIRMEM, RETURN_STATUS              ;
                               03  11 00146         BRB     23$                                         ;
                  62           54  B0 00148 22$:    MOVW    LEN, (DESCRIP)                              ;
                  6E           50  D0 0014B 23$:    MOVL    RETURN_STATUS, RETURN_STATUS                ;
                               03  11 0014E 24$:    BRB     26$                                         ; 1479
                  62           54  B0 00150 25$:    MOVW    LEN, (DESCRIP)                              ; 1500
                  50           8E  D0 00153 26$:    MOVL    RETURN_STATUS, R0                           ; 1508
                  5E           04  C0 00156         ADDL2   #4, SP                                      ; 1509
                                   05 00159         RSB                                                 ;

; Routine Size:  346 bytes,    Routine Base:  _LIB$CODE + 000E
;  321          1510  1 GLOBAL ROUTINE LIB$SFREE1_DD (          ! Deallocate a dynamic string
;  322          1511  1
;  323          1512  1         DESCRIP                         ! The descriptor to deallocate
;  324          1513  1                                 ) =
;  325          1514  1
;  326          1515  1 !++
;  327          1516  1 ! FUNCTIONAL DESCRIPTION:
;  328          1517  1 !
;  329          1518  1 !       Deallocate a string.  The string is assumed to by dynamic.
;  330          1519  1 !
;  331          1520  1 ! FORMAL PARAMETERS:
;  332          1521  1 !
;  333          1522  1 !       DESCRIP.wqu.r   The descriptor of the string to deallocate.
;  334          1523  1 !
;  335          1524  1 ! IMPLICIT INPUTS:
;  336          1525  1 !
;  337          1526  1 !       NONE
;  338          1527  1 !
;  339          1528  1 ! IMPLICIT OUTPUTS:
;  340          1529  1 !
;  341          1530  1 !       NONE
;  342          1531  1 !
;  343          1532  1 ! COMPLETION CODES:
;  344          1533  1 !
;  345          1534  1 !       SS$_NORMAL      All is OK.
;  346          1535  1 !       LIB$_INVSTRDES  Invalid string descriptor
;  347          1536  1 !       LIB$_FATERRLIB  Fatal error in the library
;  348          1537  1 !
;  349          1538  1 ! SIDE EFFECTS:
;  350          1539  1 !
;  351          1540  1 !       May deallocate virtual storage.
;  352          1541  1 !
;  353          1542  1 !--
;  354          1543  1
;  355          1544  2     BEGIN
;  356          1545  2
;  357          1546  2
;  358          1547  2 !+
;  359          1548  2 ! Free the string
;  360          1549  2 !-
;  361          1550  2     RETURN LIB$SFREE1_DD6 (.DESCRIP) ;
;  362          1551  1     END;                                        ! end of LIB$SFREE1_DD



                                          007C 00000        .ENTRY  LIB$SFREE1_DD, Save R2,R3,R4,R5,R6                    ; 1510
                        50       04   AC  D0 00002        MOVL    DESCRIP, R0                                           ; 1550
                                     0000V 30 00006        BSBW    LIB$SFREE1_DD6                                        ;
                                          04 00009        RET                                                           ; 1551

; Routine Size:  10 bytes,    Routine Base:  _LIB$CODE + 0168
```

```
1 GLOBAL ROUTINE LIB$SFREE1_DD6 (            ! Deallocate a dynamic string
1
1         DESCRIP                            ! The descriptor to deallocate
1
1                                 ) : STRING_JSB =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Deallocate a string.  The string is assumed to by dynamic.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       DESCRIP.wqu.r   The descriptor of the string to deallocate.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION CODES:
1 !
1 !       SS$_NORMAL      All is OK.
1 !       LIB$_INVSTRDES  Invalid string descriptor
1 !       LIB$_FATERRLIB  Fatal error in the library
1 !
1 ! SIDE EFFECTS:
1 !
1 !       May deallocate virtual storage.
1 !
1 !--
1
2   BEGIN
2
2   LOCAL
2       RETURN_STATUS ;
2
2   MAP
2       DESCRIP : REF BLOCK [ , BYTE] ;
2
2   !
2   ! see if this is a dynamic descriptor
2   !
2   IF .DESCRIP [DSC$B_CLASS] EQL DSC$K_CLASS_D
2   THEN
3       BEGIN
3       !
3       ! deallocate the string
3       !
4       IF (RETURN_STATUS = $STR$DEALLOCATE (DESCRIP))
3       THEN
4           BEGIN
4           !
4           ! Make sure the pointer and length field are zero, so the
```

```
;  421        1609 4              ! user is less likely to mistakenly use an old address.
;  422        1610 4              ! Also, if he calls to reallocate without reinitalizing,
;  423        1611 4              ! he will not get confused.
;  424        1612 4              !
;  425        1613 4             DESCRIP [DSC$W_LENGTH]  = 0 ;
;  426        1614 4             DESCRIP [DSC$A_POINTER] = 0 ;
;  427        1615 3             END;
;  428        1616 3         END
;  429        1617 3         !
;  430        1618 3         ! at this point, RETURN_STATUS contains the status returned
;  431        1619 3         ! by $STR$DEALLOCATE
;  432        1620 3         !
;  433        1621 2     ELSE
;  434        1622 2         !
;  435        1623 2         ! not a dynamic descriptor
;  436        1624 2         !
;  437        1625 2
;  438        1626 2         RETURN_STATUS = LIB$_INVSTRDES ;
;  439        1627 2
;  440        1628 2     RETURN .RETURN_STATUS
;  441        1629 2
;  442        1630 1     END ;                            ! of routine LIB$SFREE1_DD6



            5E               04  C2 00000 LIB$SFREE1_DD6::
                                                  SUBL2   #4, SP                                    ; 1552
            52               50  D0 00003         MOVL    R0, R2
            02         03    A2  91 00006         CMPB    3(DESCRIP), #2                            ; 1598
                             56  12 0000A         BNEQ    3$
            50 00000000G     8F  D0 0000C         MOVL    #STR$_NORMAL, RETURN_STATUS               ; 1604
            53         04    A2  D0 00013         MOVL    4(DESCRIP), R3                            ;
                             3C  13 00017         BEQL    2$                                        ;
     00F0   8F               62  B1 00019         CMPW    (DESCRIP), #240                           ;
                             1A  1A 0001E         BGTRU   1$                                        ;
            51               53  D0 00020         MOVL    R3, STRING_BLOCK                          ;
            51         FE    A1  3C 00023         MOVZWL  -2(STRING_BLOCK), ALLOC_LENGTH            ;
                             51  D7 00027         DECL    R1                                        ;
            51               07  8A 00029         BICB2   #7, R1                                    ;
            51 00000000G0041     9E 0002C         MOVAB   STR$$Q_SHORT_Q[R1], INSQUE_ADDR           ;
       00   B1               63  0E 00034         INSQUE  (R3), @0(INSQUE_ADDR)                     ;
                             1B  11 00038         BRB     2$                                        ;
                       04    A2  9F 0003A 1$:     PUSHAB  4(DESCRIP)                                ;
       04   AE               62  3C 0003D         MOVZWL  (DESCRIP), 4(SP)                          ;
                       04    AE  9F 00041         PUSHAB  4(SP)                                     ;
00000000G   00               02  FB 00044         CALLS   #2, LIB$FREE_VM                           ;
            07               50  E8 0004B         BLBS    RETURN_STATUS, 2$                         ;
            50 00000000G     8F  D0 0004E         MOVL    #STR$_FATINTERR, RETURN_STATUS            ;
            51               50  D0 00055 2$:     MOVL    RETURN_STATUS, RETURN_STATUS              ;
            0E               50  E9 00058         BLBC    RETURN_STATUS, 4$                         ;
                             62  B4 0005B         CLRW    (DESCRIP)                                 ; 1613
                       04    A2  D4 0005D         CLRL    4(DESCRIP)                                ; 1614
                             07  11 00060         BRB     4$                                        ; 1598
            51 00000000G     8F  D0 00062 3$:     MOVL    #LIB$_INVSTRDES, RETURN_STATUS            ; 1626
            50               51  D0 00069 4$:     MOVL    RETURN_STATUS, R0                         ; 1628
```

```
                                        5E            04  C0 0006C          ADDL2   #4, SP                                  ; 1630
                                                          05 0006F          RSB                                             ;

; Routine Size:  112 bytes,    Routine Base:  _LIB$CODE + 0172
;  444          1631   1 GLOBAL ROUTINE LIB$SFREEN_DD (          ! Deallocate dynamic strings
;  445          1632   1
;  446          1633   1          NUM_DESC,                      ! Number of descriptors
;  447          1634   1          DESC_PTR                       ! First descriptor to deallocate
;  448          1635   1                              ) =
;  449          1636   1
;  450          1637   1 !++
;  451          1638   1 ! FUNCTIONAL DESCRIPTION:
;  452          1639   1 !
;  453          1640   1 !        Deallocate a number of strings.  The strings are all assumed
;  454          1641   1 !        to be dynamic.
;  455          1642   1 !
;  456          1643   1 ! FORMAL PARAMETERS:
;  457          1644   1 !
;  458          1645   1 !        NUM_DESC.rl.r   The number of descriptors to deallocate.
;  459          1646   1 !        DESC_PTR.wqu.r  The first of these descriptors.
;  460          1647   1 !
;  461          1648   1 ! IMPLICIT INPUTS:
;  462          1649   1 !
;  463          1650   1 !        NONE
;  464          1651   1 !
;  465          1652   1 ! IMPLICIT OUTPUTS:
;  466          1653   1 !
;  467          1654   1 !        NONE
;  468          1655   1 !
;  469          1656   1 ! COMPLETION CODES:
;  470          1657   1 !
;  471          1658   1 !        SS$_NORMAL
;  472          1659   1 !        LIB$_FATERRLIB                   Fatal error in the library
;  473          1660   1 !
;  474          1661   1 ! SIDE EFFECTS:
;  475          1662   1 !
;  476          1663   1 !        May deallocate virtual storage.
;  477          1664   1 !
;  478          1665   1 !--
;  479          1666   1
;  480          1667   2    BEGIN
;  481          1668   2
;  482          1669   2    LIB$SFREEN_DD6 (..NUM_DESC, .DESC_PTR)
;  483          1670   2
;  484          1671   1    END;                                              ! end of LIB$SFREE1_DD



                                      007C 00000         .ENTRY  LIB$SFREEN_DD, Save R2,R3,R4,R5,R6                ; 1631
                         51      08   AC   D0 00002         MOVL    DESC_PTR, R1                                      ; 1669
                         50      04   BC   D0 00006         MOVL    @NUM_DESC, R0                                     ;
                                   0000V   30 0000A         BSBW    LIB$SFREEN_DD6                                    ;
                                           04 0000D         RET                                                       ; 1671

; Routine Size:  14 bytes,    Routine Base:  _LIB$CODE + 01E2


;  485          1672   1
```

```
GLOBAL ROUTINE LIB$SFREEN_DD6 (          ! Deallocate dynamic strings

        NUM_DESC,                        ! Number of descriptors
        DESC_PTR                         ! First descriptor to deallocate

                                ) : STRING_JSB =

!++
! FUNCTIONAL DESCRIPTION:
!
!       Deallocate a number of strings.  The strings are all assumed
!       to be dynamic.
!
! FORMAL PARAMETERS:
!
!       NUM_DESC.rl.v   The number of descriptors to deallocate.
!       DESC_PTR.wqu.r  The first of these descriptors.
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! COMPLETION CODES:
!
!       SS$_NORMAL
!       LIB$_FATERRLIB  Fatal error in the library
!
! SIDE EFFECTS:
!
!       May deallocate virtual storage.
!
!--

    BEGIN

    LOCAL
        RETURN_STATUS,
        DESCRIP : REF BLOCK [ , BYTE];

!+
! Loop through all the descriptors, freeing them.
! Quit when one fails to deallocate
!-

    INCR COUNTER FROM 1 TO .NUM_DESC DO
        BEGIN
        DESCRIP = .DESC_PTR + ((.COUNTER - 1)*DSC$K_D_BLN);

!+
! Now try freeing it.
!-
        RETURN_STATUS = LIB$SFREE1_DD6 (.DESCRIP) ;
        IF .RETURN_STATUS NEQ SS$_NORMAL
```

```
;  544          1730  3           THEN
;  545          1731  3               RETURN .RETURN_STATUS ;
;  546          1732  3
;  547          1733  2           END;    ! of INCR loop
;  548          1734  2
;  549          1735  2     !+
;  550          1736  2     ! Since we fell out of the loop above, all strings have been
;  551          1737  2     ! successfully deallocated, so...
;  552          1738  2     !-
;  553          1739  2
;  554          1740  2       RETURN (SS$_NORMAL);
;  555          1741  1       END;                                        ! end of LIB$SFREE1_DD6



                        7E          50  7D 00000 LIB$SFREEN_DD6::
                                                         MOVQ     R0, NUM_DESC                                   ; 1673
                                    7E  D4 00003         CLRL     COUNTER                                        ; 1723
                                    1F  10 00005         BSBB     2$                                             ;
            50      04  AE          03  78 00007 1$:     ASHL     #3, COUNTER, R0                                ;
                        50      0C  AE  C0 0000C         ADDL2    DESC_PTR, R0                                   ;
                        6E          70  7E 00010         MOVAQ    -(R0), DESCRIP                                 ;
                        50          6E  D0 00013         MOVL     DESCRIP, R0                                    ; 1728
                                  FF69  30 00016         BSBW     LIB$SFREE1_DD6                                 ;
                        51          50  D0 00019         MOVL     R0, RETURN_STATUS                              ;
                        01          51  D1 0001C         CMPL     RETURN_STATUS, #1                              ; 1729
                                    05  13 0001F         BEQL     2$                                             ;
                        50          51  D0 00021         MOVL     RETURN_STATUS, R0                              ; 1731
                                    09  11 00024         BRB      3$                                             ;
            DB      04  AE      08  AE  F3 00026 2$:     AOBLEQ   NUM_DESC, COUNTER, 1$                          ; 1721
                        50          01  D0 0002C         MOVL     #1, R0                                         ; 1740
                        5E          10  C0 0002F 3$:     ADDL2    #16, SP                                        ; 1741
                                        05 00032         RSB                                                     ;

; Routine Size:  51 bytes,    Routine Base:  _LIB$CODE + 01F0


;  556          1742  1
;  558          1743  1 GLOBAL ROUTINE LIB$SCOPY_DXDX (          ! Copy string by descriptor
;  559          1744  1
;  560          1745  1         SRC_DESC,                               ! Source string
;  561          1746  1         DEST_DESC                               ! Destination string
;  562          1747  1                             ) =
;  563          1748  1
;  564          1749  1 !++
;  565          1750  1 ! FUNCTIONAL DESCRIPTION:
;  566          1751  1 !
;  567          1752  1 !       Copy any supported class string passed by descriptor to any
;  568          1753  1 ! supported class string passed by descriptor.
;  569          1754  1 !
;  570          1755  1 ! FORMAL PARAMETERS:
;  571          1756  1 !
;  572          1757  1 !       SRC_DESC.rt.dx  Address of source string descriptor.
;  573          1758  1 !       DEST_DESC.wt.dx Address of destination descriptor.
;  574          1759  1 !                       The class and dtype fields are not disturbed.
;  575          1760  1 !
;  576          1761  1 ! IMPLICIT INPUTS:
;  577          1762  1 !
;  578          1763  1 !       NONE
;  579          1764  1 !
;  580          1765  1 ! IMPLICIT OUTPUTS:
;  581          1766  1 !
;  582          1767  1 !       NONE
;  583          1768  1 !
;  584          1769  1 ! COMPLETION CODES:
;  585          1770  1 !
;  586          1771  1 !       SS$_NORMAL      Success
;  587          1772  1 !
;  588          1773  1 !       LIB$_STRTRU     The source string was truncated to fit the
;  589          1774  1 !                       fixed-length destination string.
;  590          1775  1 !
;  591          1776  1 !       LIB$_INSVIRMEM  Not enough virtual memory available.
;  592          1777  1 !
;  593          1778  1 !       LIB$_INVSTRDES  Invalid DSC$B_CLASS field contents  or
;  594          1779  1 !                               If class = A or NCA, ARSIZE => 65K
;  595          1780  1 !
;  596          1781  1 ! SIDE EFFECTS:
;  597          1782  1 !
;  598          1783  1 !       May allocate and deallocate virtual storage.
;  599          1784  1 !
;  600          1785  1 !--
;  601          1786  1
;  602          1787  2    BEGIN
;  603          1788  2       RETURN LIB$SCOPY_DXDX6 (.SRC_DESC, .DEST_DESC) ;
;  604          1789  1    END;                                       ! end of LIB$SCOPY_DXDX





                                      007C 00000          .ENTRY  LIB$SCOPY_DXDX, Save R2,R3,R4,R5,R6            ; 1743
                     50        04   AC  7D 00002          MOVQ    SRC_DESC, R0                                    ; 1788
                                   0000V 30 00006          BSBW    LIB$SCOPY_DXDX6                                 ;
                                        04 00009          RET                                                     ; 1789
```

```
; Routine Size:  10 bytes,    Routine Base:  _LIB$CODE + 0223

;  605          1790  1
```

```
1 GLOBAL ROUTINE LIB$SCOPY_DXDX6 (                    ! Copy string by descriptor
1         SRC_DESC,                                   ! Source string
1         DEST_DESC                                   ! Destination string
1
1                                ) : STRING_JSB =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Copy any supported class string passed by descriptor to any
1 ! supported class string passed by descriptor.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       SRC_DESC.rt.dx  Address of source string descriptor.
1 !       DEST_DESC.wt.dx Address of destination string descriptor.
1 !                       The class and dtype fields are not disturbed.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION CODES:
1 !
1 !       SS$_NORMAL      Success
1 !
1 !       LIB$_STRTRU     The source string was truncated to fit the
1 !                       fixed-length destination string.
1 !
1 !       LIB$_INSVIRMEM  Not enough virtual memory available.
1 !
1 !       LIB$_INVSTRDES  Invalid DSC$B_CLASS field contents  or
1 !                               If class = A or NCA, ARSIZE => 65K
1 !
1 ! SIDE EFFECTS:
1 !
1 !       May allocate and deallocate virtual storage.
1 !
1 !--
1
2     BEGIN
2
2     MAP
2         SRC_DESC  : REF BLOCK [, BYTE],
2         DEST_DESC : REF BLOCK [, BYTE];
2
2 !+
2 ! Extract the length and address of 1st byte of data from the source
2 ! descriptor.  JSB to LIB$SCOPY_R_DX6 to do work.
2 !-
2     IF .SRC_DESC [DSC$B_CLASS] GTRU DSC$K_CLASS_D
2     THEN                      ! Use generalized extraction
3         BEGIN
```

```
;   664          1848  3          LOCAL
;   665          1849  3              LENGTH : VECTOR [1, LONG],  ! length of string
;   666          1850  3              DATA_ADDR : VECTOR [1, LONG] ,      ! start of data address
;   667          1851  3              RETURN_STATUS ;
;   668          1852  3
;   669          1853  3          RETURN_STATUS = LIB$ANALYZE_SDESC_R2 ( .SRC_DESC ;
;   670          1854  3                                                 LENGTH [0],
;   671          1855  3                                                 DATA_ADDR [0]) ;
;   672          1856  3
;   673          1857  3          IF NOT .RETURN_STATUS THEN RETURN (.RETURN_STATUS) ;
;   674          1858  3
;   675          1859  3          RETURN (LIB$COPY_R_DX6 ( .LENGTH, .DATA_ADDR, .DEST_DESC)) ;
;   676          1860  3
;   677          1861  3          END
;   678          1862  3
;   679          1863  2      ELSE                 ! can jsb with lenth and address directly
;   680          1864  3          BEGIN
;   681          1865  4          RETURN (LIB$COPY_R_DX6 ( .SRC_DESC [DSC$W_LENGTH],
;   682          1866  4                                   .SRC_DESC [DSC$A_POINTER],
;   683          1867  3                                   .DEST_DESC ) ) ;
;   684          1868  2          END ;
;   685          1869  2
;   686          1870  1      END;                                           ! end of LIB$COPY_DXDX6



                      53             50  7D 00000 LIB$COPY_DXDX6::
                                                          MOVQ     R0, R3                                              ; 1791
                      02         03  A3  91 00003         CMPB     3(SRC_DESC), #2                                     ; 1845
                                     1D  1B 00007         BLEQU    1$                                                  ;
                      50             53  D0 00009         MOVL     SRC_DESC, R0                                        ; 1853
                         00000000G   00  16 0000C         JSB      LIB$ANALYZE_SDESC_R2                                ;
                      56             51  D0 00012         MOVL     R1, R6                                              ;
                      55             52  D0 00015         MOVL     R2, R5                                              ;
                      18             50  E9 00018         BLBC     RETURN_STATUS, 3$                                   ; 1857
                      52             54  D0 0001B         MOVL     DEST_DESC, R2                                       ; 1859
                      51             55  D0 0001E         MOVL     DATA_ADDR, R1                                       ;
                      50             56  D0 00021         MOVL     LENGTH, R0                                          ;
                                     0A  11 00024         BRB      2$                                                  ;
                      52             54  D0 00026 1$:     MOVL     DEST_DESC, R2                                       ; 1865
                      51         04  A3  D0 00029         MOVL     4(SRC_DESC), R1                                     ;
                      50             63  3C 0002D         MOVZWL   (SRC_DESC), R0                                      ;
                                  0000V  30 00030 2$:     BSBW     LIB$COPY_R_DX6                                      ;
                                         05 00033 3$:     RSB                                                          ; 1870

; Routine Size:  52 bytes,    Routine Base:  _LIB$CODE + 022D


;   687          1871  1
;  689        1872   1 GLOBAL ROUTINE LIB$SCOPY_R_DX (          ! Copy string by reference
;  690        1873   1
;  691        1874   1        SRC_LEN,                                ! Length of source
;  692        1875   1        SRC_ADDR,                               ! Address of source data
;  693        1876   1        DEST_DESC                               ! Destination string
;  694        1877   1                              ) =
;  695        1878   1
;  696        1879   1 !++
;  697        1880   1 ! FUNCTIONAL DESCRIPTION:
;  698        1881   1 !
;  699        1882   1 !        Copy any class string passed by reference to any supported
;  700        1883   1 ! class string passed by descriptor.
;  701        1884   1 !
;  702        1885   1 ! FORMAL PARAMETERS:
;  703        1886   1 !
;  704        1887   1 !        SRC_LEN.rwu.r   Address of length of source
;  705        1888   1 !        SRC_ADDR.rt.r   Address of source
;  706        1889   1 !        DEST_DESC.wt.dx Address of destination string descriptor.
;  707        1890   1 !                        The class and dtype fields are not disturbed.
;  708        1891   1 !
;  709        1892   1 ! IMPLICIT INPUTS:
;  710        1893   1 !
;  711        1894   1 !        NONE
;  712        1895   1 !
;  713        1396   1 ! IMPLICIT OUTPUTS:
;  714        1897   1 !
;  715        1898   1 !        NONE
;  716        1899   1 !
;  717        1900   1 ! COMPLETION CODES:
;  718        1901   1 !
;  719        1902   1 !        SS$_NORMAL      Success
;  720        1903   1 !
;  721        1904   1 !        LIB$_STRTRU     The source string was truncated to fit the
;  722        1905   1 !                        fixed-length destination string.
;  723        1906   1 !
;  724        1907   1 !        LIB$_INSVIRMEM  Not enough virtual memory available.
;  725        1908   1 !
;  726        1909   1 !        LIB$_INVSTRDES  Invalid DSC$B_CLASS field contents  or
;  727        1910   1 !                                If class = A or NCA, ARSIZE => 65K
;  728        1911   1 !
;  729        1912   1 ! SIDE EFFECTS:
;  730        1913   1 !
;  731        1914   1 !        May allocate and deallocate virtual storage.
;  732        1915   1 !
;  733        1916   1 !--
;  734        1917   1
;  735        1918   2    BEGIN
;  736        1919   2    RETURN LIB$SCOPY_R_DX6 (..SRC_LEN, .SRC_ADDR, .DEST_DESC) ;
;  737        1920   1    END;                                    ! end of LIB$SCOPY_R_DX



                                  007C 00000         .ENTRY  LIB$SCOPY_R_DX, Save R2,R3,R4,R5,R6          ; 1872
                    51      08  AC  7D 00002         MOVQ    SRC_ADDR, R1                                 ; 1919
                    50      04  BC  D0 00006         MOVL    @SRC_LEN, R0                                 ;
```

```
                          0000V 30 0000A          BSBW    LIB$SCOPY_R_DX6                     ; 1920
                                04 0000D          RET

; Routine Size:  14 bytes,    Routine Base:  _LIB$CODE + 0261

;  738          1921  1
```

```
GLOBAL ROUTINE LIB$SCOPY_R_DX6 (        ! Copy string by descriptor

        SRC_LEN,                        ! Number of bytes in source
        SRC_ADDR,                       ! Address of source data
        DEST_DESC                       ! Destination string

                                ) : STRING_JSB =

!++
! FUNCTIONAL DESCRIPTION:
!
!       Copy any class string passed by reference to any supported
! class string passed by descriptor.
!
! FORMAL PARAMETERS:
!
!       SRC_LEN.rwu.v           ! (in R0) length of source
!       SRC_ADDR.rt.r           ! (in R1) pointer to source string
!       DEST_DESC.wt.dx         ! (in R2) pointer to destination
!                                         string descriptor
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! COMPLETION CODES:
!
!       SS$_NORMAL      Success
!
!       LIB$_STRTRU     The source string was truncated to fit the
!                       fixed-length destination string.
!
!       LIB$_INSVIRMEM  Not enough virtual memory available.
!
!       LIB$_INVSTRDES  Invalid DSC$B_CLASS field contents  or
!                               If class = A or NCA, ARSIZE => 65K
!
! SIDE EFFECTS:
!
!       May allocate and deallocate virtual storage.
!--

    BEGIN

    LOCAL
        RETURN_STATUS;

    MAP
        DEST_DESC : REF BLOCK [ , BYTE] , ! destination descriptor
        SRC_LEN : WORD UNSIGNED ;         ! length of input

```

```
2  !+
2  ! Select the class of descriptor.
2  ! Return the status resulting from the copy operation.
2  !-
2
2      RETURN_STATUS = SS$_NORMAL ;           ! Assume success
3      RETURN ( CASE .DEST_DESC[DSC$B_CLASS]
3                  FROM DSC$K_CLASS_Z TO DSC$K_CLASS_SB OF
3      SET
3
3  !+
3  ! fixed string descriptor (CLASS _Z, _S, _SD, _SB)
3  ! ***********************************************
3  !
3  ! Use fixed length semantics. Copy to destination with fill or
3  ! truncation.
3  !-
3          [DSC$K_CLASS_Z,
3           DSC$K_CLASS_S,
3           DSC$K_CLASS_SD,
3           DSC$K_CLASS_SB] :
4              BEGIN
4              BUILTIN R0; ! length of uncopied src from MOVC5
4
4              CH$COPY (.SRC_LEN, .SRC_ADDR, STR$K_FILL_CHAR,
4                       .DEST_DESC [DSC$W_LENGTH],
4                       .DEST_DESC [DSC$A_POINTER]); ! do copy
4
4              IF .R0 EQLU 0   ! if no uncopied src
4                  THEN
4                      SS$_NORMAL                  ! then success
4                  ELSE
4                      LIB$_STRTRU                 ! else truncation
3              END;
3
```

```
;     3  !+
;     3  ! dynamic destination string
;     3  ! **************************
;     3  !-
;     3          [DSC$K_CLASS_D] :
;     4              BEGIN
;   P 4              IF $STR$NEED_ALLOC (.SRC_LEN,
;     5                                  ($STR$DYN_AL_LEN (DEST_DESC)) )
;     5
;   L 5  %IF %BLISS (BLISS16) OR %BLISS (BLISS36)          ! if not VAX must not
;   U 5  %THEN                                             ! CH$MOVE with overlap
;   U 5  OR $STR$OVERLAP (.SRC_ADDR, .SRC_LEN,
;   U 5                    .DEST_DESC [DSC$A_POINTER], .SRC_LEN)
;     5  %FI
;     4              THEN
;     5                  BEGIN                    ! cannot directly fill dest
;     5                  LOCAL
;     5                      LOC_RET_STAT,        ! status of calls to Allocate
;     5                                           !  and Deallocate
;     5                      TEMP_DESC : $STR$DESCRIPTOR;        ! create temp
;     5
;     5                      LOC_RET_STAT = $STR$ALLOCATE (.SRC_LEN, TEMP_DESC);
;     5                                                           ! alloc temp
;     5                      !+
;     5                      ! Allocate will only return STR$_NORMAL or
;     5                      ! STR$_INSVIRMEM, therefore if it wasn't success,
;     5                      ! don't continue copying
;     5                      !-
;     5
;     6                      IF (.LOC_RET_STAT)
;     5                          THEN
;     6                              BEGIN        ! successful allocate
;     6                              CH$MOVE (.SRC_LEN, .SRC_ADDR, ! copy to temp
;     6                                        .TEMP_DESC [DSC$A_POINTER]);
;     6                              $STR$EXCH_DESCS (TEMP_DESC, DEST_DESC);
;     6                                                           ! switch temp
;     6                                                           ! and dest
;     6                              LOC_RET_STAT = $STR$DEALLOCATE (TEMP_DESC);
;     6                                                           ! return former
;     6                                                           ! string
;     6                              !+
;     6                              ! $STR$DEALLOCATE returns either STR$_NORMAL
;     6                              ! or STR$_FATINTERR.
;     6                              !-
;     6                              IF NOT .LOC_RET_STAT
;     6                              THEN
;     6                                  RETURN_STATUS = LIB$_FATERRLIB ;
;     6                              END          ! successful allocate
;     5                          ELSE
;     5                              RETURN_STATUS = LIB$_INSVIRMEM ;
;     5                  END                      ! cannot directly fill dest
;     5
;     4              ELSE
;     4
;     5                  BEGIN                    ! directly fill dest
;     5                  CH$MOVE (.SRC_LEN, .SRC_ADDR,   ! write dest
;     5                            .DEST_DESC [DSC$A_POINTER]);
;  890          2070  5               DEST_DESC [DSC$W_LENGTH] = .SRC_LEN;
;  891          2071  4               END;                    ! directly fill dest
;  892          2072  4
;  893          2073  4           .RETURN_STATUS                      ! return the status
;  894          2074  3           END;
;  895          2075  3
```

```
!+
! Class A and NCA array descriptor
! ********************************
!-
        [DSC$K_CLASS_A,         ! Class A Array descriptor
         DSC$K_CLASS_NCA]:      ! Class NCA array descriptor
            BEGIN
            BUILTIN R0; ! len of uncopied src from MOVC5

            IF .DEST_DESC [DSC$L_ARSIZE] GTR MAX_SIZE ! If size>max
            THEN LIB$_INVSTRDES ;                      ! then quit

            CH$COPY (.SRC_LEN, .SRC_ADDR, STR$K_FILL_CHAR,
                     .DEST_DESC [DSC$L_ARSIZE],
                     .DEST_DESC [DSC$A_POINTER]); ! do copy

            IF .R0 EQLU 0   ! if no uncopied src
            THEN
                RETURN_STATUS = SS$_NORMAL      ! then success
            ELSE
                RETURN_STATUS = LIB$_STRTRU   !else truncation

            END ;                 ! of Class A and NCA Array Descriptor
```

```
;  921        2099  3  !+
;  922        2100  3  ! Varying string descriptor
;  923        2101  3  ! **************************
;  924        2102  3  !-
;  925        2103  3
;  926        2104  3             [DSC$K_CLASS_VS]:       ! Varying string descriptor
;  927        2105  4                BEGIN
;  928        2106  5                IF (.SRC_LEN LEQU .DEST_DESC [DSC$W_MAXSTRLEN] )
;  929        2107  4                THEN        ! fits within MAXLEN, copy and update CURLEN
;  930        2108  5                    BEGIN
;  931        2109  5                    CH$MOVE (.SRC_LEN, .SRC_ADDR,
;  932        2110  5                            .DEST_DESC [DSC$A_POINTER] + 2);
;  933        2111  5                    (.DEST_DESC [DSC$A_POINTER])<0,16> = .SRC_LEN ;
;  934        2112  5                    SS$_NORMAL        ! return success status
;  935        2113  5                    END
;  936        2114  5
;  937        2115  4                ELSE        ! Won't fit within MAXLEN. Only copy MAXLEN's
;  938        2116  4                            ! worth of data and update CURLEN to MAXLEN
;  939        2117  4
;  940        2118  5                    BEGIN
;  941        2119  5                    CH$MOVE (.DEST_DESC [DSC$W_MAXSTRLEN], .SRC_ADDR,
;  942        2120  5                            .DEST_DESC [DSC$A_POINTER] + 2);
;  943        2121  5                    (.DEST_DESC [DSC$A_POINTER])<0,16> =
;  944        2122  5                                        .DEST_DESC [DSC$W_MAXSTRLEN] ;
;  945        2123  5                    LIB$_STRTRU       ! return truncation status
;  946        2124  5                    END
;  947        2125  5
;  948        2126  3                END ;                 ! of Varying string descriptor
;  949        2127  3
;  950        2128  3
;  951        2129  3  !+
;  952        2130  3  ! Unsupported class descriptor
;  953        2131  3  ! ****************************
;  954        2132  3  !-
;  955        2133  3
;  956        2134  3           [INRANGE, OUTRANGE]:     ! Unsupported class of descriptor
;  957        2135  3               LIB$_INVSTRDES ;
;  958        2136  2     TES);                          ! end of set on class code
;  959        2137  2
;  960        2138  1     END;                                             ! end of LIB$SCOPY_R_DX6


                                                        .EXTRN  STR$$MOVQ_R1

                 5E          1C  C2 00000 LIB$SCOPY_R_DX6::
                                                        SUBL2   #28, SP                          ; 1922
             04  AE          52  D0 00003               MOVL    R2, DEST_DESC                    ;
                             51  DD 00007               PUSHL   R1                               ;
             04  AE          50  D0 00009               MOVL    R0, SRC_LEN                      ;
                 56          01  D0 0000D               MOVL    #1, RETURN_STATUS                ; 1983
         7E  08  AE          03  C1 00010               ADDL3   #3, DEST_DESC, -(SP)             ; 1984
         0F      00          9E  8F 00015               CASEB   @(SP)+, #0, #15                  ;
   0020     004F     0029   0029    00019 1$:           .WORD   3$-1$,-                          ;
   0020     0020     0020   020D    00021                       3$-1$,-                          ;
   022F     020D     0029   0020    00029                       7$-1$,-                          ;
   0029     0020     0020   0020    00031                       2$-1$,-                          ;
```

```
                                                                          34$-1$,-                                    :
                                                                          2$-1$,-                                     :
                                                                          2$-1$,-                                     :
                                                                          2$-1$,-                                     :
                                                                          2$-1$,-                                     :
                                                                          3$-1$,-                                     :
                                                                          34$-1$,-                                    :
                                                                          35$-1$,-                                    :
                                                                          2$-1$,-                                     :
                                                                          2$-1$,-                                     :
                                                                          2$-1$,-                                     :
                                                                          3$-1$                                       :
                           56 00000000G    8F   D0 00039 2$:      MOVL    #LIB$_INVSTRDES, R6                         :
                                           23   11 00040          BRB     6$                                          :
                 7E      08   AE           04   C1 00042 3$:      ADDL3   #4, DEST_DESC, -(SP)                        : 2004
                                           9E   DD 00047          PUSHL   @(SP)+                                      :
 0C   BE         20      04   BE      08   AE   2C 00049          MOVC5   SRC_LEN, @SRC_ADDR, #32, @DEST_DESC, @(SP)+ :
                                           9E      00051                                                              :
                                           50   D5 00052          TSTL    R0                                          : 2006
                                           05   12 00054          BNEQ    4$                                          :
                              50           01   D0 00056          MOVL    #1, R0                                      :
                                           07   11 00059          BRB     5$                                          :
                              50 00000000G 8F   D0 0005B 4$:      MOVL    #LIB$_STRTRU, R0                            :
                              56           50   D0 00062 5$:      MOVL    R0, R6                                      :
                                         0217   31 00065 6$:      BRW     39$                                         :
                 51      08   AE           04   C1 00068 7$:      ADDL3   #4, DEST_DESC, R1                           : 2020
                              50           61   D0 0006D          MOVL    (R1), R0                                    :
                                           51   D4 00070          CLRL    R1                                          :
                                           50   D5 00072          TSTL    R0                                          :
                                           06   12 00074          BNEQ    8$                                          :
                                           51   D6 00076          INCL    R1                                          :
                                           52   D4 00078          CLRL    R2                                          :
                                           15   11 0007A          BRB     10$                                         :
                       00F0   8F      08   BE   B1 0007C 8$:      CMPW    @DEST_DESC, #240                            :
                                           06   1B 00082          BLEQU   9$                                          :
                              52      08   BE   3C 00084          MOVZWL  @DEST_DESC, R2                              :
                                           07   11 00088          BRB     10$                                         :
                              52           50   D0 0008A 9$:      MOVL    R0, STRING_BLOCK                            :
                              52      FE   A2   3C 0008D          MOVZWL  -2(STRING_BLOCK), R2                        :
                   000000F0   8F           52   D1 00091 10$:     CMPL    R2, #240                                    :
                                           26   1F 00098          BLSSU   14$                                         :
                              04           51   E9 0009A          BLBC    R1, 11$                                     :
                                           52   D4 0009D          CLRL    R2                                          :
                                           15   11 0009F          BRB     13$                                         :
                       00F0   8F      08   BE   B1 000A1 11$:     CMPW    @DEST_DESC, #240                            :
                                           06   1B 000A7          BLEQU   12$                                         :
                              52      08   BE   3C 000A9          MOVZWL  @DEST_DESC, R2                              :
                                           07   11 000AD          BRB     13$                                         :
                              52           50   D0 000AF 12$:     MOVL    R0, STRING_BLOCK                            :
                              52      FE   A2   3C 000B2          MOVZWL  -2(STRING_BLOCK), R2                        :
      52         04   AE      10           00   ED 000B6 13$:     CMPZV   #0, #16, SRC_LEN, R2                        :
                                           26   13 000BC          BEQL    18$                                         :
                                           27   11 000BE          BRB     19$                                         :
                              04           51   E9 000C0 14$:     BLBC    R1, 15$                                     :
                                           52   D4 000C3          CLRL    R2                                          :
                                           15   11 000C5          BRB     17$                                         :
                       00F0   8F      08   BE   B1 000C7 15$:     CMPW    @DEST_DESC, #240                            :
```

```
                                      06  1B 000CD        BLEQU    16$
                        52         08 BE  3C 000CF        MOVZWL   @DEST_DESC, R2
                                      07  11 000D3        BRB      17$
                        52            50  D0 000D5 16$:   MOVL     R0, STRING_BLOCK
                        52         FE A2  3C 000D8        MOVZWL   -2(STRING_BLOCK), R2
52    04  AE            10            00  ED 000DC 17$:   CMPZV    #0, #16, SRC_LEN, R2
                                      03  1A 000E2        BGTRU    19$
                                    0132  31 000E4 18$:   BRW      33$
                        07  00000000G 00  E8 000E7 19$:   BLBS     STR$$V_INIT, 20$                 2034
              00000000G 00            00  FB 000EE        CALLS    #0, STR$$INIT
                        50  00000000G 8F  D0 000F5 20$:   MOVL     #STR$_NORMAL, RETURN_STATUS
                   00F0 8F         04 AE  B1 000FC        CMPW     SRC_LEN, #240
                                      43  1A 00102        BGTRU    26$
                                   04 AE  B5 00104        TSTW     SRC_LEN
                                      04  12 00107        BNEQ     21$
                                      53  D4 00109        CLRL     TEMP
                                      31  11 0010B        BRB      25$
                        51         04 AE  3C 0010D 21$:   MOVZWL   SRC_LEN, R1
                                      51  D7 00111        DECL     R1
                        51            07  8A 00113        BICB2    #7, R1
                        54  00000000G0041  9E 00116        MOVAB    STR$$Q_SHORT_Q[R1], REMQUE_ADDR
                        53         00 B4  0F 0011E 22$:   REMQUE   @0(REMQUE_ADDR), TEMP
                                      05  1D 00122        BVS      23$
                        52            01  D0 00124        MOVL     #1, ALLOC_DONE
                                      0D  11 00127        BRB      24$
                                      52  D4 00129 23$:   CLRL     ALLOC_DONE
                        7E         04 AE  3C 0012B        MOVZWL   SRC_LEN, -(SP)
              00000000G 00            01  FB 0012F        CALLS    #1, STR$$ALOC_SHORT
                        05            52  E8 00136 24$:   BLBS     ALLOC_DONE, 25$
                        2E            50  E9 00139        BLBC     RETURN_STATUS, 28$
                                      E0  11 0013C        BRB      22$
                        29            50  E9 0013E 25$:   BLBC     RETURN_STATUS, 28$
                     1C AE            53  D0 00141        MOVL     TEMP, TEMP_DESC+4
                                      1E  11 00145        BRB      27$
                                   1C AE  9F 00147 26$:   PUSHAB   TEMP_DESC+4
                     10 AE         08 AE  3C 0014A        MOVZWL   SRC_LEN, 16(SP)
                                   10 AE  9F 0014F        PUSHAB   16(SP)
              00000000G 00            02  FB 00152        CALLS    #2, LIB$GET_VM
                        09            50  E8 00159        BLBS     RETURN_STATUS, 27$
                        50  00000000G 8F  D0 0015C        MOVL     #STR$_INSVIRMEM, RETURN_STATUS
                                      05  11 00163        BRB      28$
                     18 AE         04 AE  B0 00165 27$:   MOVW     SRC_LEN, TEMP_DESC
                     0C AE            50  D0 0016A 28$:   MOVL     RETURN_STATUS, LOC_RET_STAT
                        03         0C AE  E8 0016E        BLBS     LOC_RET_STAT, 29$                2042
                                    009B  31 00172        BRW      32$
      1C  BE         00 BE         04 AE  28 00175 29$:   MOVC3    SRC_LEN, @SRC_ADDR, @TEMP_DESC+4 2046
                     10 AE         08 BE  B0 0017C        MOVW     @DEST_DESC, $STR$TEMP_DESC       2047
          50         08 AE            04  C1 00181        ADDL3    #4, DEST_DESC, R0
                     14 AE            60  D0 00186        MOVL     (R0), $STR$TEMP_DESC+4
          50         08 AE            02  C1 0018A        ADDL3    #2, DEST_DESC, R0
                     1A AE            60  90 0018F        MOVB     (R0), TEMP_DESC+2
          50         08 AE            03  C1 00193        ADDL3    #3, DEST_DESC, R0
                     1B AE            60  90 00198        MOVB     (R0), TEMP_DESC+3
                        50         18 AE  9E 0019C        MOVAB    TEMP_DESC, R0
                        51         08 AE  D0 001A0        MOVL     DEST_DESC, R1
                            00000000G 00  16 001A4        JSB      STR$$MOVQ_R1
                     18 AE         10 AE  B0 001AA        MOVW     $STR$TEMP_DESC, TEMP_DESC
```

```
                          1C   AE        14   AE  D0 001AF        MOVL     $STR$TEMP_DESC+4, TEMP_DESC+4              ;
                               50 00000000G   8F  D0 001B4        MOVL     #STR$_NORMAL, RETURN_STATUS                ; 2050
                               52        1C   AE  D0 001BB        MOVL     TEMP_DESC+4, R2                            ;
                                              3E  13 001BF        BEQL     31$                                        ;
                        00F0   8F        18   AE  B1 001C1        CMPW     TEMP_DESC, #240                            ;
                                              1A  1A 001C7        BGTRU    30$                                        ;
                               51             52  D0 001C9        MOVL     R2, STRING_BLOCK                           ;
                               51        FE   A1  3C 001CC        MOVZWL   -2(STRING_BLOCK), ALLOC_LENGTH             ;
                                              51  D7 001D0        DECL     R1                                         ;
                               51             07  8A 001D2        BICB2    #7, R1                                     ;
                               51  00000000G0041  9E 001D5        MOVAB    STR$$Q_SHORT_Q[R1], INSQUE_ADDR            ;
                          00   B1             62  0E 001DD        INSQUE   (R2), @0(INSQUE_ADDR)                      ;
                                              1C  11 001E1        BRB      31$                                        ;
                                         1C   AE  9F 001E3 30$:   PUSHAB   TEMP_DESC+4                                ;
                          0C   AE        1C   AE  3C 001E6        MOVZWL   TEMP_DESC, 12(SP)                          ;
                                         0C   AE  9F 001EB        PUSHAB   12(SP)                                     ;
                   00000000G   00             02  FB 001EE        CALLS    #2, LIB$FREE_VM                            ;
                               07             50  E8 001F5        BLBS     RETURN_STATUS, 31$                         ;
                               50 00000000G   8F  D0 001F8        MOVL     #STR$_FATINTERR, RETURN_STATUS             ;
                          0C   AE             50  D0 001FF 31$:   MOVL     RETURN_STATUS, LOC_RET_STAT                ;
                               78        0C   AE  E8 00203        BLBS     LOC_RET_STAT, 39$                          ; 2057
                               56 00000000G   8F  D0 00207        MOVL     #LIB$_FATERRLIB, RETURN_STATUS             ; 2059
                                              6F  11 0020E        BRB      39$                                        ; 2042
                               56 00000000G   8F  D0 00210 32$:   MOVL     #LIB$_INSVIRMEM, RETURN_STATUS             ; 2062
                                              66  11 00217        BRB      39$                                        ; 2019
                 60        00   BE        04   AE  28 00219 33$:   MOVC3    SRC_LEN, @SRC_ADDR, (R0)                   ; 2069
                          08   BE        04   AE  B0 0021F        MOVW     SRC_LEN, @DEST_DESC                        ; 2070
                                              59  11 00224        BRB      39$                                        ; 2073
                 50        08   AE             0C  C1 00226 34$:   ADDL3    #12, DEST_DESC, R0                         ; 2085
                 7E        08   AE             04  C1 0022B        ADDL3    #4, DEST_DESC, -(SP)                       ; 2090
                                              9E  DD 00230        PUSHL    @(SP)+                                     ;
                 7E        0C   AE             0C  C1 00232        ADDL3    #12, DEST_DESC, -(SP)                      ;
    9E           20        08   BE        0C   AE  2C 00237        MOVC5    SRC_LEN, @SRC_ADDR, #32, @(SP)+, @(SP)+    ;
                                              9E     0023E                                                            ;
                                              50  D5 0023F        TSTL     R0                                         ; 2092
                                              35  12 00241        BNEQ     38$                                        ;
                               56             01  D0 00243        MOVL     #1, RETURN_STATUS                          ; 2094
                                              1D  11 00246        BRB      36$                                        ;
                 51        08   AE             04  C1 00248 35$:   ADDL3    #4, DEST_DESC, R1                          ; 2110
                               50             61  9E 0024D        MOVAB    (R1), R0                                   ;
                          08   BE        04   AE  B1 00250        CMPW     SRC_LEN, @DEST_DESC                        ; 2106
                                              13  1A 00255        BGTRU    37$                                        ;
                               56             60  D0 00257        MOVL     (R0), R6                                   ; 2110
            02   A6        00   BE        04   AE  28 0025A        MOVC3    SRC_LEN, @SRC_ADDR, 2(R6)                  ;
                               66        04   AE  B0 00261        MOVW     SRC_LEN, (R6)                              ; 2111
                               56             01  D0 00265 36$:   MOVL     #1, R6                                     ; 2108
                                              15  11 00268        BRB      39$                                        ;
                               56             60  D0 0026A 37$:   MOVL     (R0), R6                                   ; 2120
            02   A6        00   BE        08   BE  28 0026D        MOVC3    @DEST_DESC, @SRC_ADDR, 2(R6)               ;
                               66        08   BE  B0 00274        MOVW     @DEST_DESC, (R6)                           ; 2122
                               56 00000000G   8F  D0 00278 38$:   MOVL     #LIB$_STRTRU, R6                           ; 2118
                               50             56  D0 0027F 39$:   MOVL     R6, R0                                     ; 1984
                               5E             20  C0 00282        ADDL2    #32, SP                                    ; 2138
                                                  05 00285        RSB                                                 ;

; Routine Size:  646 bytes,    Routine Base:  _LIB$CODE + 026F
```

```
;  962          2139  1 END
;  963          2140  0 ELUDOM




;                                       PSECT SUMMARY
;
;       Name                       Bytes                         Attributes
;
;    _LIB$CODE                          1269  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)




;                                Library Statistics
;
;                                            -------- Symbols --------        Pages        Processing
;       File                                 Total    Loaded   Percent        Mapped       Time
;
;    _$255$DUA28:[SYSLIB]STARLET.L32;1        9776        16         0          581          00:00.8




;                                        COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:LIBSCOPY/OBJ=OBJ$:LIBSCOPY MSRC$:LIBSCOPY/UPDATE=(ENH$:LIBSCOPY)

; Size:          1269 code + 0 data bytes
; Run Time:         00:18.3
; Elapsed Time:     01:17.2
; Lines/CPU Min:     7012
; Lexemes/CPU-Min: 32267
; Memory Used:  205 pages
; Compilation Complete
```

0209 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

LIBPOLYG
LIS

LIBREMQHI
LIS

LIBSIGSTO
LIS

LIBRENAME
LIS

LIBSCANC
LIS

LIBRDOBJ
LIS

LIBRUNPRO
LIS

LIBSIGNAL
LIS

LIBPUTOUT
LIS

LIBREMQTI
LIS

LIBSIGRET
LIS

LIBSIMTRA
LIS

LIBSCOPY
LIS

LIBPOLYH
LIS

LIBREVERT
LIS